**Amendments to the Claims**:

The listing of claims will replace all prior versions, and listings, of claims in the application:

**Listing of Claims**:

1. (Currently Amended) A method for operating a software module on a processor unit in a controller networked via a data bus in a vehicle, wherein i) the software module is executable in a plurality of controllers which interchange data via the data bus, ii) selection of the controller on which the software module is operated is made based on the available computational capacity of the controllers which are currently in operation, and iii) each of the controllers can turn off the software module when a utilization level of its processor is high, and as soon as the software module has been turned off, the software module is to be started again on another controller; said method comprising:

a software module, in a controller on which said software module is running, sending to the data bus, either cyclically or upon request, an appropriate identifier containing information indicating the software module's ~~its~~ operating status and the identity of the controller on which ~~it~~ the software module is running;

checking cyclically to determine whether and on which controller the software module is running, based on said identifier; and determining which of the controllers has the greatest free computation capacity, taking into account its processor clock frequency;

wherein said determining step is made based on information sent by ~~virtue of~~ the controllers, ~~involved sending~~ in rotation or by means of a request, wherein the information ~~that~~ is indicative of their available computational capacity.

2. (Previously Presented) The method as claimed in claim 1, wherein before the software module is executed it is ascertained which of the controllers provides the maximum free computation capacity and the software module is started on the determined controller.

3. (Previously Presented) The method as claimed in claim 1, wherein the controller on which the software module is running compares its computation capacity with the computation capacity of the other controllers and either continues to operate or terminates operation of the software module based on the comparison.

4. (Previously Presented) The method as claimed in claim 1, wherein the computation capacity of a controller is ascertained from the processor utilization level and processor type.

5. (Previously Presented) The method as claimed in claim 1, wherein the software module is started on a controller having the maximum free computation capacity.

6. (Previously Presented) The method as claimed in claim 1, wherein the software module is stored in a memory in the controllers.

7. (Canceled)

8. (Currently Amended) A networked controller having software modules stored in a controller's memory; wherein:

the software modules perform primary ~~control~~ tasks;

a software module with a ~~subsidiary~~ secondary task can be additionally stored in a microcontroller’s memory by the controllers;

the controllers have process cycles;

a process cycle is terminated after a particular time has elapsed, the data ascertained in the process are output onto the data bus, and the process cycle is started again;

the process cycle for the controllers is determined by the software modules for one of the primary tasks ~~task~~, the operating system and a bus protocol; and

when a process cycle or a process cycle time has elapsed, data ~~are~~ is sent to the data bus which characterize their current processor utilization level and processor type used, with the controllers using ~~these~~ this data to ascertain the utilization level of the other controllers.

9. (Currently Amended) A method of operating a network of controllers which are coupled via a data bus, each of which controllers has at least one processor, and has installed thereon the same software module which can be executed by the processor contained in any one of the controllers, each of said controllers being configured such that it can turn off the software module when a utilization of its processor is high, said method comprising:

each controller sending via the data bus, information regarding a current utilization level of its at least one processor;

whenever said software module is running in a particular one of said controllers, said software module in said particular controller sending via the data bus, an identifier indicating ~~it's~~ the software module's operating state and identifying the particular controller;

checking said data bus to determine whether an identifier is present;

~~if~~ when no identifier is found in said checking step, determining which of the controllers has the greatest available computation capacity, based on its current utilization level as sent via the data bus;

said controller with said greatest available utilization level starting operation of said software module, and said software module sending to said data bus, an identifier indicating ~~it's~~ the software module's operating status and the identity of the controller in which ~~it~~ the software module is running;

~~if~~ when an identifier is present on the data bus in said checking step, the controller on which said software module is running ascertaining its own processor utilization level and comparing its computation capacity with the available computational capacity of other controllers coupled via the data bus;

~~if~~ when the utilization level of the controller on which the software module is greater than that of one of said other controllers, said controller on which said software module is running ceasing operation of said software module; and said one of said other controllers starting operation of said software module, and said software module sending to said data bus an identifier indicating that it is running and identifying said one controller.

10. (New) The networked controller of claim 8, wherein the software module sends via the data bus, an identifier indicating the software module's operating state and identifying the particular controller on which the software module is running.